漫谈Java加密技术Java认证考试 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/629/2021_2022__E6_BC_AB_E8_B0_88Java_c104_629764.htm 接下来我们介绍对称加密算法，最常用的莫过于DES数据加密算法。 DES DES-Data Encryption Standard，即数据加密算法。是IBM公司于1975年研究成功并公开发表的。DES算法的入口参数有三个：Key、Data、Mode.其中 Key为8个字节共64位，是DES算法的工作密钥；Data也为8个字节64位，是要被加密或被解密的数据；Mode为DES的工作方式，有两种：加密或解密。 DES算法把64位的明文输入块变为64位的密文输出块，它所使用的密钥也是64位。 通过java代码实现如下 import java.security.Key. import java.security.SecureRandom. import javax.crypto.Cipher. import javax.crypto.KeyGenerator. import javax.crypto.SecretKey. import javax.crypto.SecretKeyFactory. import javax.crypto.spec.DESKeySpec. /** *//** * DES安全编码组件 author by http://www.bt285.cn http://www.5a520.cn * * gt. * 支持DES、DESede(TripleDES,就是3DES)、AES、Blowfish、RC2、RC4(ARCFOUR) * DES key size must be equal to 56 * DESede(TripleDES) key size must be equal to 112 or 168 * AES key size must be equal to 128, 192 or 256,but 192 and 256 bits may not be available * Blowfish key size must be multiple of 8, and can only range from 32 to 448 (inclusive) * RC2 key size must be between 40 and 1024 bits * RC4(ARCFOUR) key size must be between 40 and 1024 bits * 具体内容 需要关注 JDK Document http:///docs/technotes/guides/security/SunProviders.html * gt. * *

@author 梁栋 * @version 1.0 * @since 1.0 */ public abstract class DESCoder extends Coder { /** *//** * ALGORITHM 算法 gt. * 可替换为以下任意一种算法，同时key值的size相应改变。 * * gt. * DES key size must be equal to 56 * DESede(TripleDES) key size must be equal to 112 or 168 * AES key size must be equal to 128, 192 or 256,but 192 and 256 bits may not be available * Blowfish key size must be multiple of 8, and can only range from 32 to 448 (inclusive) * RC2 key size must be between 40 and 1024 bits * RC4(ARCFOUR) key size must be between 40 and 1024 bits * gt. * * 在Key toKey(byte[] key)方法中使用下述代码 * gt.SecretKey secretKey = new SecretKeySpec(key, ALGORITHM).gt. 替换 * gt. * DESKeySpec dks = new DESKeySpec(key). * SecretKeyFactory keyFactory = SecretKeyFactory.getInstance(ALGORITHM). * SecretKey secretKey = keyFactory.generateSecret(dks). * gt. */ public static final String ALGORITHM = "DES". /** *//** * 转换密钥gt. * * @param key * @return * @throws Exception */ private static Key toKey(byte[] key) throws Exception { DESKeySpec dks = new DESKeySpec(key). SecretKeyFactory keyFactory = SecretKeyFactory.getInstance(ALGORITHM). SecretKey secretKey = keyFactory.generateSecret(dks). // 当使用其他对称加密算法时，如AES、Blowfish等算法时，用下述代码替换上述三行代码 // SecretKey secretKey = new SecretKeySpec(key, ALGORITHM). return secretKey. } /** *//** * 解密 * * @param data * @param key * @return * @throws Exception */ public static byte[] decrypt(byte[] data, String key) throws Exception { Key k = toKey(decryptBASE64(key)). Cipher cipher =

Cipher.getInstance(ALGORITHM). cipher.init(Cipher.DECRYPT_MODE, k). return cipher.doFinal(data). } /** *//** * 加密 * * @param data * @param key * @return * @throws Exception */ public static byte[] encrypt(byte[] data, String key) throws Exception { Key k = toKey(decryptBASE64(key)). Cipher cipher = Cipher.getInstance(ALGORITHM). cipher.init(Cipher.ENCRYPT_MODE, k). return cipher.doFinal(data). } /** *//** * 生成密钥 * * @return * @throws Exception */ public static String initKey() throws Exception { return initKey(null). } /** *//** * 生成密钥 * * @param seed * @return * @throws Exception */ public static String initKey(String seed) throws Exception { SecureRandom secureRandom = null. if (seed != null) { secureRandom = new SecureRandom(decryptBASE64(seed)). } else { secureRandom = new SecureRandom(). } KeyGenerator kg = KeyGenerator.getInstance(ALGORITHM). kg.init(secureRandom). SecretKey secretKey = kg.generateKey(). return encryptBASE64(secretKey.getEncoded()). } } 延续上一个类的实现，我们通过MD5以及SHA对字符串加密生成密钥，这是比较常见的密钥生成方式。 再给出一个测试类： import static org.junit.Assert.*. import org.junit.Test. /** *//** * * @author by http://www.bt285.cn http://www.5a520.cn * @version 1.0 * @since 1.0 */ public class DESCoderTest { @Test public void test() throws Exception { String inputStr = "DES". String key = DESCoder.initKey(). System.err.println("原文:\t" inputStr).

System.err.println("密钥:\t" key). byte[] inputData = inputStr.getBytes(). inputData = DESCoder.encrypt(inputData, key). System.err.println("加密后:\t" DESCoder.encryptBASE64(inputData)). byte[] outputData = DESCoder.decrypt(inputData, key). String outputStr = new String(outputData). System.err.println("解密后:\t" outputStr). assertEquals(inputStr, outputStr). } } 得到的输出内容如下： 原文 ： DES 密钥 ： f3wEtRrV6q0= 加密后 ： C6qe9oNIzRY= 解密后 ： DES 由控制台得到的输出，我们能够比对加密、解密后结果一致。这是一种简单的加密解密方式，只有一个密钥。 其实DES有很多同胞兄弟，如DESede（TripleDES）、AES 、Blowfish、RC2、RC4（ARCFOUR）。这里就不过多阐述了，大同小异，只要换掉ALGORITHM换成对应的值，同时做一个代码替换SecretKey secretKey = new SecretKeySpec（key, ALGORITHM）；就可以了，此外就是密钥长度不同了。 /** * DES key size must be equal to 56 * DESede(TripleDES) key size must be equal to 112 or 168 * AES key size must be equal to 128, 192 or 256,but 192 and 256 bits may not be available * Blowfish key size must be multiple of 8, and can only range from 32 to 448 (inclusive) * RC2 key size must be between 40 and 1024 bits * RC4(ARCFOUR) key size must be between 40 and 1024 bits **/